1. Expectations
    a. Goal for the workshop is to deploy ALZ reference architecture based on the Azure Barista scenario. You are then free to move on to optionally side quest based on your liking.

2. Clone ALZ-Bicep Repo to your own Github or locally to your computer. You can also create your own fork repository from main.
    a. Which repo?
        i. Azure/ALZ-Bicep: This repository contains the Azure Landing Zones (ALZ) Bicep modules that help deliver and deploy the Azure Landing Zone conceptual architecture in a modular approach. https://aka.ms/alz/docs (github.com)
        ii. Go through ways of consuming: ConsumerGuide · Azure/ALZ-Bicep Wiki (github.com)

3. Look into the deployment flow:
    a. DeploymentFlow · Azure/ALZ-Bicep Wiki (github.com)
    b. Go through the Module Deployment Sequence and the orchestration modules

4. Go through each module/orchestration thoroughly and start filling out the parameter file in your repository.
    a. Remember to look at the customer requirements
    b. If you have only 2 subscriptions for this hackathon, use same subscription for networking and log analytics workspace/automation account and place it under Connectivity Management Group
        i. Hint if using pipeline, this can easily be configured in the variables section
    c. Try to understand what each module does.
    d. Module 6: Role Assignments, will be used in a later stage and is not needed in the initial deployment
    e. Module 9: Spoke Networking, can be used to deploy a spoke network for identity subscriptions where domain controllers can reside. We will use the vending module later to handle Landing Zone virtual network creation

5. Make pull-request/commit with the applied changes

6. You are now ready to deploy! We would like to challenge you to construct a pipeline following our guide but if you would like to deploy each module from your local client you are free to do so.

If you choose to deploy from local client (otherwise skip to DevOps section):

Look at deployment section of each parameter file for a code snippet on how to deploy each module in Powershell or Az CLI.

DevOps: Create initial pipeline for deployment

Azure baristas want to leverage Github repository in order to manage their platform changes. This enable them to have version control and single pane of glass to all changes being made. In order to deploy to Azure from the repo, pipelines need to be created through Github Actions.

1. Create the initial pipeline and .yaml file.
    a. PipelinesOverview · Azure/ALZ-Bicep Wiki (github.com)
    b. Does the pipeline example make sense? **Are there any modules missing compared to the deployment flow?** How does it differentiate from the deployment flow? Have they swapped out orchestrations for modules?
        i. Why do they create resource groups?
        ii. Why do they have variables stated in the pipeline?
        iii. Does it make sense to have several pipelines for different deployments?
    c. Create a .yaml file under .github/workflows/xxx.yaml take inspiration from the example in .b

2. Setup Deployment Identity (SPN) and Service Connection (Azure DevOps) / Connect Github Actions to Azure (Github)
    a. https://github.com/Azure/ALZ-Bicep/wiki/DeploymentFlow#deployment-identity
    b. To get secret information:
        i. alz-partner-lab/createspnandsetpermission.ps1 at main · reduards/alz-partner-lab (github.com)
    c. Remember to put the Github secret in following format, name in example as AZURE_CREDENTIALS:

```
{
  "clientId": "<GUID>",
  "clientSecret": "<GUID>",
  "subscriptionId": "<GUID>",
  "tenantId": "<GUID>"
}
```

3. Time to deploy ALZ-Bicep by running the workflow, will it shine?
    a. Confirm all findings in the portal, what have been deployed at what scope?

Automation: Deploy Vending Module for Application Landing Zone creation

Azure Barista want to automate their creation of their application landing zone. They have decided that they will gather input from Service Now and they will have a line of approval for production subscriptions. They are now looking for ways of automating their deployments by leveraging git. Since they already are using Bicep for their platform deployment they have come across the ALZ Bicep Vending Module and now want to implement it.

Setup Landing Zone Module and Create your first Landing Zone: Azure/bicep-lz-vending: Bicep module & pipelines to deploy landing zone subscriptions (github.com)

1. Go through first page and look at the what automation will be taken care of. Is there something missing? Will perhaps policies handle some of these automation pieces?
2. Decide how you will consume the module, new repo? Same repo? New Pipeline? Locally? ConsumerGuide · Azure/bicep-lz-vending Wiki (github.com)
3. Once you are ready to consume the module. Create your first parameter file. Choose your way of deploying it. Either locally or via pipeline from a Github repository.
4. Create parameter file under a map called “landingZones”.
    a. './landingZones/lz1.parameters.json'
    b. Look at the example for how the parameter file should be constructed. In this case we are moving a subscription.
5. To see how the vending module is deployed look at the example on first page.
6. Time to deploy! Similar to initial deployment. We challenge you to deploy via pipeline in Github Actions but if you would like to deploy the module locally via Az CLI or Powershell you are free to do so.
    a. Create a .yaml file under .github/workflows/xxx.yaml take inspiration from the example from example in 5.
    b. Remember we are moving an existing subscription so we do not need any permissions to be delegated to the SPN for subscription creation in EA. However, this will be the case for many customer deployments.
    c. When you are done constructing the pipeline, go ahead and run it. Run workflow!

7. If you manage to have a successful run. Look in the portal to confirm the creation of the landing zone and its deployments.
8. Try to deploy something to the landing zone so you can see how the policy is are taking into effect
    a. Deploy a VM and look at the diagnostic settings and backups. Have they been enabled?
    b. Now try to deploy a PaaS service like Azure Key Vault with private endpoints. Remember deploy without your own Private DNS Zone to look and see if the record get registered in the central zone.
    c. Look at the other default policies and feel free to try out any other expected behaviour

Some other aspects to have in mind going forward

How will this scale? Will we deploy every parameter file every time? How do we handle changes to existing landing zones? Look into issues at Github, some requested features on the way

What do we need to add to the vending machine? What should be the process when creating a new landing zone?

    a. IPAM: Azure/ipam: IP Address Management on Azure (github.com)
    b. Frontend? Service Now? How do we send in the parameters to Azure DevOps? What should be the input?
    c. Line of approval?
    d. Pre-done templates

Side Quests

Great you are done with the initial deployment! You can now chose any of the side quests depending on your liking. Good Luck!

- **DevOps:** Pipeline Deployment (if not done during initial deployment)
- **Identity & Access management:** Enable access rights for Platform team
- **Security:** Change policy assignments to adhere to stricter requirements

Or look into our Enterprise perspectives

- **Test and Development**
- **Protecting your branch and repo**

Identity & Access management: Enable access rights for Platform team

Azure baristas have now identified that they want to leverage the custom roles to give just enough access to their different platform roles. To begin with they need to create the groups which will then be populated with members.

1. Create 4 security groups one for NetOps, one for SecOps, one for AppOps and Platform Owner(Owner). This can be done via portal or Powershell since it is not supported in bicep yet. Take not of the object ID since it will be used as a parameter later.
2. Use the Role Assignment module to delegate respective role assignments for the AD Security Groups leveraging the custom roles deployed previously.
    a. Find role ID via portal or via Powershell.
    b. Make sure to fill in the right parameter file.
    c. Role assignment can be made at sudo root (contoso) level
3. Save/commit the new changes and deploy the new module
4. Visually confirm in portal that the roles have been delegated at correct scope

Security: Change policy assignments to adhere to stricter requirements

Azure Baristas have noticed that there is no policy enforcing next hop to the Azure Firewall in the central hub for corp landing zones. This means that they cant be certain East-West traffic between vnets and north-south traffic to on-prem will bypass the central firewall. They noticed that there is a custom policy defined called Deny-Subnet-Without-Udr. They now want to add this policy as an assignment for corp landing zone to enforce the use of route table at subnet creation.

1. Look at the policy deep dive to learn more of how to make policy assignments. Choose one of the approaches and edit the module accordingly.
2. We can recommend to follow the guide for updating existing alzdefaultassignments
3. **Notice how *Deny-Subnet-Without-Udr* already exist in the *assignment lib* as well as in the *_policyAssignmentsBicepInput.txt* file. Just need to add the necessary rows in the bicep file: alzDefaultPolicyAssignments.bicep**
4. When necessary changes have been made deploy the module.
5. Visually confirm at the Corp Management Group that the assignment have been made.

Enterprise Perspective

If you have a completed all side challenges, go ahead and implement following enterprise perspectives

1. Look into how you will protect your repo and what the process should look like to contribute and do changes
    a. Branch Policies
    b. Forbid direct commit to the main branch?
    c. How shall Pull Request be handled? Squash merge?
    d. How shall testing and validation be handled?
    e. Git branch policies and settings - Azure Repos | Microsoft Learn
    f. Merge strategies and squash merge - Azure Repos | Microsoft Learn

2. Look into the process of how you will be testing new changes
    a. Development lifecycle - Cloud Adoption Framework | Microsoft Learn
    b. Environments - Cloud Adoption Framework | Microsoft Learn
    c. Test-driven development for Azure Landing Zones - Cloud Adoption Framework | Microsoft Learn
    d. Testing approach for Azure landing zones - Cloud Adoption Framework | Microsoft Learn
    e. Do you need a new SPN for testing purposes? Discuss a bit regarding which approach that make sense and what needs to be done.
    f. Deploy your first landing zone, look in the portal to confirm